A Novel Cost Optimization Strategy for SDN-enabled UAV-assisted Vehicular Computation Offloading

Liang Zhao, *Member, IEEE,* Kaiqi Yang, Zhiyuan Tan, *Member, IEEE,* Xianwei Li, Suraj Sharma, and Zhi Liu, *Senior Member, IEEE*

***Abstract*—Vehicular computation offloading is a well-received strategy to execute delay-sensitive and/or compute-intensive tasks of legacy vehicles. The response time of vehicular computation offloading can be shortened by using mobile edge computing that offers strong computing power, driving these computation tasks closer to end users. However, the quality of communication is hard to guarantee due to the obstruction of dense buildings or lack of infrastructure in some zones. Unmanned Aerial Vehicles (UAVs), therefore, have become one of the means to establish communication links for the two ends owing to its characteristics of ignoring terrain and flexible deployment. To make a sensible decision of computation offloading, nevertheless vehicles need to gather offloading-related global information, in which Software-Defined Networking (SDN) has shown its advances in data collection and centralized management. In this paper, thus, we propose an SDN-enabled UAV-assisted vehicular computation offloading optimization framework to minimize the system cost of vehicle computing tasks. In our framework, the UAV and the Mobile Edge Computing (MEC) server can work on behalf of the vehicle users to execute the delay-sensitive and compute-intensive tasks. The UAV, in a meanwhile, can also be deployed as a relay node to assist in forwarding computation tasks to the MEC server. We formulate the offloading decision-making problem as a multi-players computation offloading sequential game, and design the UAV-assisted Vehicular Computation Cost Optimization (UVCO) algorithm to solve this problem. Simulation results demonstrate that our proposed algorithm can make the offloading decision to minimize the Average System Cost (ASC).**

***Index Terms*—Mobile edge computing, computation offloading, vehicular networks, unmanned aerial vehicles, game theory.**

I. Introduction

VEHICULAR network (VN) is developed as a new networking paradigm to enable the data transmissions among vehicles for allowing the control and management of urban traffic and providing intelligent guidance and service for vehicles in recent years. With this new paradigm, the information transmission and network access can be realized through vehicle-to-everything (V2X), including vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication, where such technologies have been designed in forms of 5G, DSRC [1] or VANET [2], [3]. However, VN has still faced great challenges like dynamic topology, stability, and deployment. For example, VNs are susceptible to the dynamic road conditions including highly-mobile vehicles and changing of vehicle density, in which both may result in high packet loss and latency. Consequently, as an emerging network paradigm, Software-Defined Vehicular Network (SDVN) has been starting to attract research attention [4], [5], [6], [7], [8], [9] as one of the most important technologies to manage VNs. Similar to the general softwarized network, SDVN separates the data plane and control plane in which the data plane is mainly composed of vehicles, infrastructure; the control plane is a logic controller that provides services for the unit of the data plane with its global view of the network. These distinguishing characteristics of the SDVN architecture make VN more flexible and easier to implement intelligent schemes.

With the promotion of VNs, the Internet of Vehicles (IoV) has been developed rapidly [10], [11], [12] to bring us various vehicular applications, such as route planning, autonomous driving, and infotainment applications [13]. These applications can either ensure travel safety or provide the entertainment interconnection in the journey. We can observe that most of these applications are delay-sensitive and resource-intensive, with complex computation and high energy demand. However, at present, many legacy vehicles are still of limited storage capacity and insufficient computing resources. Therefore, it is important to take advantage of SDVN to offload complex computation tasks of legacy vehicles to execute such tasks somewhere else.

Mobile Edge Computing (MEC) is an ideal technology for the above where it integrates capabilities, including collection, calculation, and storage. Distinguished from cloud computing [14], MEC can respond faster as it is closer to users[15]. Compared with the vehicles, MEC servers are equipped with rich computing resources, which can provide efficient computing services for the users in the coverage. Hence, vehicles can choose to offload the computation tasks to the MEC server through the wireless connection in which the MEC server can execute the computation tasks on behalf of vehicle users in order to enhance the capabilities of vehicle users handling the resource-intensive applications and improve computing efficiency [16], [17], [18], [19], [20], [21]. However, if we intend to apply offloading computation technology in the real

Liang Zhao (lzhao@sau.edu.cn) and Kaiqi Yang (kaiqiyangkq@163.com) are with School of Computer Science, Shenyang Aerospace University, China.

Zhiyuan Tan (z.tan@napier.ac.uk) is with School of Computing, Edinburgh Napier University, UK; He is the corresponding author.

Xianwei Li (xianweili@fuji.waseda.jp) is with School of Information Engineering, Suzhou University, China; He is the corresponding author.

Suraj Sharma (suraj@iiit-bh.ac.in) is with Department of Computer Science and Engineering, International Institute of Information Technology Bhubaneswar, India.

Zhi Liu (liu@shizuoka.ac.jp) is with Department of Mathematical and Systems Engineering, Shizuoka University, Japan.

applications, it will face more challenges such as network accessibility and obstacles. Regions can be with no access point or base stations to allow network access in these costly infrastructures. Also, the dense buildings can block the signal off transmissions. As a result, these constraints can limit the use of the applications without guaranteeing the reliability of offloading computation.

On the other hand, in recent years, Unmanned Aerial Vehicles (UAVs) have been widely studied in military and civilian fields benefited from its characteristics such as flexible mobility, low environmental requirements, and variety of sizes [22], [23], [24], [25], [26], [27]. In networking, UAVs can be dispatched as aerial Base Stations (BSs), Access Points (APs) or relays, which therefore, assist the terrestrial communication and computing. Vehicle users can choose to offload tasks to UAVs built with computing resource for execution. Unfortunately, as a side effect, offloading computation tasks can generate additional cost on execution time and energy consumption from sending offloading data to returning computation results. Minimizing these costs is the key to implement the offloading computations.

In this paper, we propose a novel Software-Defined Network (SDN)-enabled UAV-assisted vehicular computation offloading cost optimization framework. This framework enables the vehicles to execute computationally complex and time-sensitive computation tasks while minimizing the task execution time and system energy consumption. With building our framework in the concept of SDN, the controller can ensure the information exchange by collecting global information and sending them to each vehicle. In particular, a UAV with rich computing resources is employed to assist legacy vehicles to perform delay-sensitive and compute-intensive tasks in a zone with dense buildings and a limited infrastructure hosting only one MEC server.

Computation offloading for legacy vehicles is a promising technology, however, among which making the optimal offloading decision is the key challenge to realize the MEC. We adopt the idea of game theory to address this challenge. As such, vehicle users can make mutually satisfactory decision-making according to their own interests, and reduce the pressure of central management. Moreover, in existing work [28], smart devices such as UAV and Vehicle can only execute one computation task at a time. As a solution, we set up a waiting queue for the UAV and the MEC server according to the principle of First-Come-First-Serve (FCFS) queue to store computation tasks temporarily that are later offloaded to improve the efficiency of resource utilization. In addition, we also set up a forwarding queue for UAV based on the FCFS scheduling principle to solve the situation that multiple vehicles have forwarding requirements at the same time. We further propose an algorithm to achieve the Nash equilibrium to minimize the execution time and energy consumption.

The main contributions of this paper are as follows.

- **SDN-enabled UAV-assisted vehicular computing or relay**:We deploy a UAV equipped with intelligent computing capabilities to cruise along fixed trajectories in the region with dense buildings and sparse infrastructure. In this scenario, vehicle users can offload computation tasks to the UAV, which can execute the computation tasks on behalf of the vehicle users. In addition, the UAV can serve as a relay node to transfer the offloaded computation tasks of vehicle users to the MEC server.
- **UAV-assisted vehicular computation cost optimization algorithm**: We formulate the UAV-assisted vehicular computation offloading decision-making problem as multi-vehicle users offloading game and design a decision-making algorithm. The objective is to optimize the execution time of the vehicular computation task, system energy consumption, in order to ensure the QoS of communication links.
- **Dynamic scenario**: Our scenario is more realistic. Different from existing work such as [14], we consider the dynamic nature of vehicle which allows vehicles to move in our scenario. In addition, the UAV cruises at a fixed trajectory and altitude in our scenario.

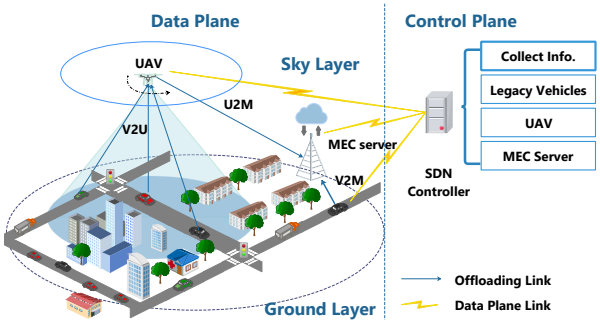

Fig. 1. The proposed offloading architecture

The rest of the paper is organized as follows. In Section II, we introduce the system model of our proposal and take communication and computation aspects into account. The multi-player computation offloading game and our proposed UVCO algorithm are described in detail in Section III. Simulation results are shown in Section IV. We conclude the paper in Section V.

II. System Model

In this section, we introduce the system model of legacy vehicles offloading computation tasks. The system model is shown in Fig. 1. The controller is deployed to collect the real-time information (coordinate position, speed, waiting queue length, etc.) of the global devices (the legacy vehicles, the UAV, and the MEC server) in the data plane. This controller also sends the information to each device in the data plane to realize the information exchange in the coverage area of the SDN controller. We consider a set $N = \{1, 2, 3..., a\}$ of legacy vehicles that have insufficient computation power. In addition, these legacy vehicles will generate a set of computing-intensive and delay-sensitive computation tasks denoted by I. The computation tasks have their own priorities, representing the degree of the execution time demand. The higher the priority of the task, the faster it is required to be executed. There is one MEC server that can execute computation tasks on behalf of the vehicle users. However, due to the dynamic

TABLE I
COMPARISON WITH EXISTING WORK

Related work	Computing source	SDN based	Dynamic	Delay	ASC
[14]	Cloud	No	No	Low	Medium
[16]	MEC	No	No	High	Medium
[18]	MEC	No	Yes	High	Medium
[19]	MEC	No	No	Medium	High
[20]	MEC	No	No	Medium	High
[28]	Smart vehicle	No	No	High	Low
UVCO	UAV MEC	Yes	Yes	Low	Low

movements of the vehicles in the building intensive scenario, the vehicle signal cannot transmit through the tall buildings. Therefore, the occlusion of the buildings may make it impossible for the vehicles and the MEC server to establish a wireless connection directly. Hence, we deploy a UAV as a cloudlet or a relay node to improve the computation efficiency of legacy vehicles. A coordinate system is established according to the origin at the lower-left corner of the road. The distance between the UAV and a vehicle within its coverage is calculated by the Pythagorean theorem. Most of the existing studies [25], [29] on UAV-assisted mobile device computation offloading are assumed in the static scenario. Nevertheless, without considering the dynamic nature of the vehicle, the existing assumption is not realistic. In this paper, we propose a scenario in which vehicle users move dynamically. Then, since both the communication and computation are key to vehicular computing, we will introduce the two models and the payoff function to calculate the cost in detail. In addition, the main parameters of this paper are shown in Table II.

A. Channel Model

In our scenario, vehicle user n $(n \in N)$ intends to offload tasks to the device (UAV or the MEC server) that can establish a communication link (i.e., depends on whether the communication link is blocked by obstacles). We denote $S_N = \{0, 1, 2, 3\}$ as a set of the decision-making by the user n. We have $S_n = 0$ if the vehicle user n chooses to execute a computation task locally; $S_n = 1$ if n chooses to offload the computation task to UAV; $S_n = 2$ if n chooses to offload the computation task to MEC server; and $S_n = 3$ if n chooses to offload the computation task to the MEC server through the UAV relay. We define the link-state as Non-Line of Sight (NLoS) when the communication link between two devices is blocked by the obstacles. Otherwise, it is defined as Line of Sight (LoS). In this paper, we assume that the connection between a Vehicle to the UAV (V2U) is LoS. Therefore, we can formulate the data transmission rate of offloading the task from n to the UAV via wireless links as [14]

$$R_{n,u} = Wlog_2(1 + \frac{P_V \rho_{LoS} d_{n,u}^{-\alpha_V}}{N_0 + \sum_{s \in N, s \neq n} P_V \rho_{LoS} d_{s,u}^{-\alpha_V}}) \quad (1)$$

where ρ_{LoS} is the shadow fading component following the Gamma distribution; $d_{n,u}$ denotes the distance between n and the UAV; and α_V is the path loss exponent for V2I channels; W is the channel bandwidth, and P_V is the transmission power of the vehicle; N_0 represents the noise power; $\sum_{s \in N, s \neq n} P_V \rho_{LoS} d_{s,u}^{-\alpha_V}$ represents the vehicle users (excluding n) choosing to offload computation tasks to the UAV via the wireless links simultaneously. Then, the data transmission rate of offloading task from n to the MEC server via wireless links as

$$R_{n,M} = Wlog_2(1 + \frac{P_V \rho_{v,M} d_{n,M}^{-\alpha_V}}{N_0 + \sum_{s \in N, s \neq n} P_V \rho_{v,M} d_{s,M}^{-\alpha_V}}) \quad (2)$$

where $\rho_{v,M}$ denotes the small-scale fading; $d_{n,M}$ is the distance between the vehicle user n and the MEC server; $\sum_{s \in N, s \neq n} P_V \rho_{v,M} d_{s,M}^{-\alpha_V}$ is the vehicle users (excluding n) who choose to offload computation tasks to the MEC server simultaneously via the wireless links. In this scenario, the UAV and vehicles move dynamically. Therefore, the transmission rate will vary regarding the changing distance between the two devices. From Eq. (1) and (2), if too many vehicle users choose to offload computation tasks via the wireless links simultaneously, it may cause severe interference. The value of $\sum_{s \in N, s \neq n} P_V \rho_{v,M} d_{s,M}^{-\alpha_V}$ will increase. Then the data transmission rate will be greatly lowered, and the efficiency of offloading computation tasks via wireless links will be reduced. In this circumstance, we define the link-state as Weak LoS (WLoS).

B. Computation Model

We consider that vehicle user n will generate a computation task $I_n = (C_i, O_i, D_i), I_n \in I$ in a certain time interval needs to be executed in time. Here, C_i denotes the total Central Processing Unit (CPU) cycles required to accomplish the task I_n; O_i is the total size of offloading computation task data (i.e., the input parameters and the program codes); and D_i represents the size of the execution results of task returned to the requesting vehicle. In our scenario, the legacy vehicles travel in a dense building area and with less infrastructure. In order to improve the computation efficiency of the legacy vehicles for ensuring the quality of communication service, we employ a UAV to assist legacy vehicles. On the other hand, the lifetime of UAV power supply is limited. Therefore, in the follows, we will focus on discussing the computation cost in terms of the execution time of the vehicle and the system energy consumption for each strategy.

1) Local Computing: For the local computing strategy, a vehicle user executes the computation task locally. The execution time is related to the computation capability (i.e., the CPU cycles per second) $F_{CPU}^{V,n}$ of vehicle user n, where each vehicle is of limited computation capability which may also vary. The computation execution time for each vehicle user to perform a task $I_n, (I_n \in I)$ locally is given as

$$T_{I_n}^{Loc} = \frac{C_i}{F_{CPU}^{V,n}} \quad (3)$$

The total energy consumption of task I_n executed locally in vehicle n $(E_{I_n}^{Loc})$ can be given as

$$E_{I_n}^{Loc} = C_i \times e_{CPU}^{V} \quad (4)$$

TABLE II
NOTATIONS

Notation	Description
γ_n^T/γ_n^E	Weight of execution time / energy consumption in the payoff function
$R_{n,u}/R_{n,M}$	The transmission rate of data sent by vehicle n to UAV / MEC server
$\rho_{Los}/\rho_{v,M}$	Component of shadow fading / small-scale fading
$d_{n,u}/d_{n,M}$	Distance between vehicle n and the UAV / MEC server
α_V	Path loss exponent for V2I channels
W	Channel bandwidth
N_0	Noise power
P_V	The transmission power of vehicles
$C_i/O_i/D_i$	Total CPU cycles required for execution / offload data size / size of execution results of computation task I_n
$F_{CPU}^{V,n}/F_{CPU}^{MEC}/F_{CPU}^{UAV}$	Computation capability of the vehicle user n / MEC server / UAV
$e_{CPU}^{UAV}/e_{CPU}^{V}$	Energy consumed in each CPU cycle of the UAV / vehicles executing locally
$e_{SEND}^{V}/e_{SEND}^{UAV}$	Energy consumed by a vehicle / UAV to send one data unit to the other devices.

where e_{CPU}^V denotes the energy consumed in each CPU cycle of a vehicle for computing locally. Then, we calculate the overhead of the local computing where this payoff function can be given as

$$L_{I_n,V}^{Loc} = \gamma_n^T T_{I_n}^{Loc} + \gamma_n^E E_{I_n}^{Loc} \tag{5}$$

where γ_n^T and γ_n^E denote the weight of execution time and energy consumption, respectively; the value of γ_n^T depends on the priority value of the computation task; and $\gamma_n^T + \gamma_n^E = 1$. In order to satisfy the user-specific demands, we use a weighted payoff function to adapt scenarios under different requirements. If the user's energy level is insufficient, the user can set greater weight for energy consumption during the decision-making to save energy. In contrast, suppose that the user intends to execute a delay-sensitive computation task (such as collision monitoring which requires in real-time process) in another scenario, the user can increase the weight of the execution time to reduce delay.

2) Direct Offloading to MEC server: When a connection to the MEC server can be established, vehicle user n may decide to offload the computation task I_n to the MEC server and then the MEC server will execute I_n on behalf of n. Under these circumstances, the total execution time of the task I_n is the sum of the execution time of the task I_n on the MEC server, the duration of user n transmitting the data packets containing its task I_n, and the duration of the executed results return to n. Since the MEC server is far away from the vehicle, the data sent by the server to the UAV will not be interfered with by the signal of the vehicle, while the LTE interface is adopted for communication. As a result, we can formulate the execution time of the task I_n for offloading to the MEC server as

$$T_{I_n}^{V-M} = T_{V-M}^S + T_{I_n}^{MEC} + T_{M-V}^R \tag{6}$$

where

$$T_{V-M}^S = \frac{O_i}{R_{n,M}} \tag{7}$$

$$T_{I_n}^{MEC} = \frac{C_i}{F_{CPU}^{MEC}} \tag{8}$$

$$T_{M-V}^R = \frac{D_i}{R_{LTE}} \tag{9}$$

where R_{LTE} denotes the data transmission rate of LTE interface; F_{CPU}^{MEC} is the computing frequency of the MEC server, which is the number of CPU cycles per second. We assume that the energy consumption of executing tasks by the MEC server and sending the results by the MEC server is not considered. Then we can give the total energy consumption of vehicle user n offloading the computation task I_n to MEC server as

$$E_{I_n}^{V-M} = O_i \times e_{SEND}^V \tag{10}$$

where e_{SEND}^V denotes the energy consumed by a vehicle to send one data unit to the other devices. Then, the payoff function of the vehicle user n offloading computation task I_n to the MEC server directly strategy can be given as

$$L_{I_n}^{V-M} = \gamma_n^T T_{I_n}^{V-M} + \gamma_n^E E_{I_n}^{V-M} \tag{11}$$

3) Offloading to UAV: This strategy can be achieved in two ways, in which both are based on the premise that the vehicle user can establish a connection with the UAV. The first is that the vehicle user n offloads the computation tasks to the UAV; then the UAV will execute the task I_n on behalf of n. The second is to offload the computation tasks to the UAV; then, the UAV is deployed as a relay node to assist in forwarding the tasks to the MEC server; the MEC server will execute the task I_n on behalf of n and send back the results of computation via the UAV to n. Next, we will introduce them in detail.

(a) The UAV executes the computation tasks instead of vehicle users

When a link can be established between the UAV and the vehicle user n, the task I_n can be offloaded from n to the UAV. Once tasks are received, the UAV will execute the task on behalf of n.

In this way, the total execution time of the task I_n will include the time duration of the user n transmitting the packet containing the task I_n to the UAV, the time duration of executing the computation task I_n in the UAV, and the time duration of the computation results sent back to the user n. Since only one UAV is deployed in this scenario, the data sent by the UAV to other devices will not be interfered with by the signal, and the communication adopts the Wi-Fi interface. The total execution time of offloading computation task I_n to the UAV can be given that

$$T_{I_n}^{V-U} = T_{V-U}^S + T_{I_n}^{UAV} + T_{U-V}^R \tag{12}$$

where

$$T_{V-U}^S = \frac{O_i}{R_{n,u}} \tag{13}$$

$$T_{I_n}^{UAV} = \frac{C_i}{F_{CPU}^{UAV}} \tag{14}$$

$$T_{U-V}^{R} = \frac{D_i}{R_{WiFi}} \tag{15}$$

where R_{WiFi} represents the data transmission rate of the Wi-Fi interface; F_{CPU}^{UAV} denotes the computation frequency of the UAV, which is the number of the CPU cycles per second. Due to the limited power life of the UAV, the energy consumption of the UAV is a key challenge to realize the applications of the UAV. In this way, the total energy consumption of offloading the computation task I_n to the UAV is the energy of sending the computation task to the UAV plus the energy of executing I_n by the UAV plus the energy of the results sending back from the UAV. Consequently, the total energy consumption of offloading computation task I_n to the UAV is given as

$$E_{I_n}^{V-U} = E_{V-U}^{S} + E_{I_n}^{UAV} + E_{U-V}^{R} \tag{16}$$

where

$$E_{V-U}^{S} = O_i \times e_{SEND}^{V} \tag{17}$$

$$E_{I_n}^{UAV} = C_i \times e_{CPU}^{UAV} \tag{18}$$

$$E_{U-V}^{R} = D_i \times e_{SEND}^{UAV} \tag{19}$$

where e_{CPU}^{UAV} denotes the energy consumed in each CPU cycle of the UAV executing locally; e_{SEND}^{UAV} is the energy consumption of the UAV via the Wi-Fi interface for sending one data unit to n. Then, we can formulate the payoff function of offloading the computation task I_n to UAV for executing I_n on behalf of the vehicle user n as

$$L_{I_n}^{V-U} = \gamma_n^T T_{I_n}^{V-U} + \gamma_n^E E_{I_n}^{V-U} \tag{20}$$

(b) UAV as a relay node

In this way, the vehicle user n chooses to offload the computation task I_n to the UAV. Then, the UAV relays I_n to the MEC server. After that, the MEC server will execute I_n on behalf of n. The total execution time can be defined as the sum of the time interval user n sending I_n to the UAV, the time interval of the UAV relaying I_n to the MEC server, the time interval of the MEC server executing I_n, the time interval of the computation results sent back to the UAV, and the time interval from the UAV to the user n. Therefore, the total execution time of the UAV as a relay node to execute I_n generated by n can be given as

$$T_{I_n}^{V-U-M} = T_{V-U}^{S} + T_{U-M}^{S} + T_{I_n}^{MEC} + T_{M-U}^{R} + T_{U-V}^{R} \tag{21}$$

where

$$T_{U-M}^{S} = \frac{O_i}{R_{WiFi}} \tag{22}$$

$$T_{M-U}^{R} = \frac{D_i}{R_{LTE}} \tag{23}$$

In addition, the total energy consumption is the sum of the energy of sending the data packets, including the task I_n to the UAV, the energy of the UAV sending task I_n to the MEC server, and the energy sending the computation results to the requester n by the UAV. Then the total energy consumption can be given by

$$E_{I_n}^{V-U-M} = E_{V-U}^{S} + E_{U-M}^{S} + E_{U-V}^{R} \tag{24}$$

where

$$E_{U-M}^{S} = O_i \times e_{SEND}^{UAV} \tag{25}$$

Then, we can give the payoff function of the way that the UAV is deployed as the relay node

$$L_{I_n}^{V-U-M} = \gamma_n^T T_{I_n}^{V-U-M} + \gamma_n^E E_{I_n}^{V-U-M} \tag{26}$$

In summary, the vehicle users need to select the most beneficial one of these strategies to execute the computation tasks. In the next section, we will adopt the game theory method to make the vehicle users calculate the optimal offloading decision.

III. MULTI-PLAYER COMPUTATION OFFLOADING GAME

In this section, we formulate the computation offloading decision-making problem of legacy vehicles as a multi-player computation tasks offloading game. Game is the interaction between rational and intelligent players. Here, the rationality means players choose their own strategies to maximize utility, in which intelligence represents that players can calculate their optimal strategy independently. The behavior of the players conforms to their interest, where no one has the motivation to deviate unilaterally. Hence, game theory is a powerful framework to analyze the interaction between players.

Throughout our study, we tackle the issue that how vehicle users choose the efficient and green strategy to minimize the cost of executing the delay-sensitive and computational complex tasks. This cost includes the time to execute the computation task generated by the legacy vehicle user and the system energy consumption. The main reason for adopting game theory to address offloading decision-making problems is that it can make decentralized and low-complexity offloading decisions. Vehicle users choose to maximize their benefits and calculate the optimal decision independently. At the same time, it can reduce the workload of the SDN controller in computing and sending offloading decisions. The controller only needs to collect global information and send it to users.

A. Game Formulation

In UAV-assisted VNs with dense buildings and less infrastructure, the offloading decision-making problem can be formulated as a perfect information sequential game of R players. By deploying the controller to distributing the global information, all players will know the statuses (i.e., parameters) of the system, where this can reduce the delay of information acquisition. Players take satisfactory solutions regarding their interests mutually. The sequential game can be indicated as $G(R, S_r, L)$ and next three elements of the game will be introduced.

1) Players: The players of the game are the users of legacy vehicles who need to make computation offloading decisions, can be represented as a set $R = \{1, 2, ..., r\}$.

2) Strategies: The set of strategies for the players can be represented as $S_r = \{s_1, s_2, ..., s_n, ..., s_r\}$, where $s_n \in \{0, 1, 2, 3\}$.

3) Payoff function: Let L denote the payoff function of the computation task I_n of each player in the sequential game. Payoff depends on execution time and energy consumption. The payoff function of the player n who chooses the strategy $s_n(s_n \in S_r)$ is

$$L_n^{s_n} = \lim_{\theta \to 0^+} \theta^{|s_n - 0|} L_{I_n,V}^{Loc} + \lim_{\theta \to 0^+} \theta^{|s_n - 1|} L_{I_n}^{V-U} + \lim_{\theta \to 0^+} \theta^{|s_n - 2|} L_{I_n}^{V-M} + \lim_{\theta \to 0^+} \theta^{|s_n - 3|} L_{I_n}^{V-U-M} (s_n \in \{0,1,2,3\}) \tag{27}$$

where θ is constant. The decision made by other players except player n is denoted as a set $s_{-n} = (s_1, ..., s_{n-1}, s_{n+1}, ..., s_r)$. Then, a player can choose a strategy based on minimization of their costs, i.e.,

$$\min_{s_n \in \{0,1,2,3\}} L_n(s_n, s_{-n}), n \in N$$

According to Eq. (27), we can obtain the payoff function of player n as

$$L_n(s_n, s_{-n}) = \begin{cases} L_{I_n,V}^{Loc} & if\ s_n = 0 \\ L_{I_n}^{V-U} & if\ s_n = 1 \\ L_{I_n}^{V-M} & if\ s_n = 2 \\ L_{I_n}^{V-U-M} & if\ s_n = 3 \end{cases} \tag{28}$$

The decision-making of each player is sequential, which means the player who acts later can know the choice of the player who acts earlier. Consequently, we call the game G as a multi-player computation offloading sequential game. $L_n(s_n, s_{-n})$ is the payoff function of the vehicle user n which aims to calculate the cost of executing the computation task I_n. Next, we will introduce the concept of Nash equilibrium (NE), and investigate the existence of the NE.

Definition 1. Given a strategic game $G(R, S_r, L)$ of the offloading computation game and its strategy group $S_r^* = \{s_1^*, s_2^*, ..., s_n^*, ..., s_r^*\}$, if

$$L_n(s_n^*, s_{-n}) \leq L_n(s_n, s_{-n}), \forall s_n \in S_r, n \in N \tag{29}$$

here S_r^* is a NE of the computation offloading game G.

When the game is at NE, no one can reduce its cost by unilaterally changing its strategy. Then, we can define S_r^* as an optimal strategy set. The optimal strategy (i.e., the way players execute computation tasks) is formulated by optimizing the execution time and the energy consumption of executing tasks.

According to the above definition, we can see that all the users make the corresponding optimal decisions at NE according to the decision of other participants. In the current state, all players are in the most favorable states.

B. Decentralized Computation Offloading Mechanism

In this part, we will introduce the decentralized computation offloading of the multi-player computation offloading game. The decentralized computation offloading mechanism mainly composed of two parts, judging connectivity and judging the optimal utility.

TABLE III
THE POSSIBLE STRATEGY SET FOR VEHICLES

Strategy set	Connectible to MEC server	Not connectible to MEC server
Connectible to UAV	Locally, V-M, V-U, V-U-M	Locally, V-U, V-U-M
Not Connectible to UAV	Locally, V-M	Locally

1) Judging Connectivity:

(a) The Connectivity Between Vehicle and MEC server

For a vehicle, we first consider the possibility of establishing a communication link by sending a request message to the MEC server. If the connections can be established, vehicles then consider whether to offload the computation task to the MEC server. On the contrary, if the connection fails, vehicles only consider whether offload the tasks to the UAV or execute the task locally.

(b) Whether the vehicle is within the UAV communication coverage

Vehicles should determine whether they are within the communication range of the UAV. In our scenario, we set the flight path of the UAV to cruise in a fixed trajectory to ensure that most of the vehicles in the scenario are covered by the communication range of the UAV as far as possible. However, due to the limited communication power, there still will be vehicles outside the communication coverage of the UAV. These vehicles broadcast the requests continuously to determine whether they can establish communication with the UAV. When legacy vehicles are within the communication range of the UAV, the vehicles can offload tasks to the UAV. Otherwise, if the reply packet has not been received after the timeout, the vehicle is out of the UAV communication range.

2) Decision making by judging the optimal utility:

Assuming that vehicle user n has a computation task I_n needs to be processed. Based on the part of judging connectivity, if the UAV or the MEC server can be connected to n, it will affect the decision types of n. If neither can be connected, the vehicle user has to execute the computation task locally. In the proposed framework, the SDN controller distributes the network information to the vehicles in real-time including the locations of the UAV and the MEC server, the number of tasks in the waiting queue of the UAV and the MEC server, the computing power of the UAV and the MEC server, and the number of vehicles offloading data to the UAV or the MEC server at the same time. Then, the vehicle user will make the offloading decision based on the global information provided by the SDN controller. Moreover, the decision-making of executing the computation task of the vehicle user depends on the value of the payoff function. According to Eq. (27) and Eq. (28), we introduce the strategy of selecting the value of the payoff function. In light of the value of each strategy's payoff function, the strategy with the lowest cost is selected as the optimal decision.

Next, we will introduce the process of choosing the strategy to execute the computation task by maximizing computing

Algorithm 1: UAV-assisted Vehicle Computing Cost Optimization

Input: Each vehicle user n generates a set of time-sensitive computation tasks I_n;
Output: optimal offloading decision scheme.

Initialization: Each vehicle user n first chooses the computation decision local computation: $A = 0$ and make the initial optimal decision $s_n = 0$
compute the initial value of local computation payoff function $\mu_n^{A=0}$.
measure the transmission rate of vehicle user n to the UAV and vehicle user n to the MEC server and the expected waiting time to offload to the UAV or MEC server waiting queue.
repeat
  for each vehicle user n and each computation task execution time slot t in parallel:
  if *vehicle user n can be connected to the MEC server* **then**
    if *vehicle user n can be connected to the UAV* **then**
      compute the value of payoff function $\mu_n^{A=1}$, $\mu_n^{A=2}$and $\mu_n^{A=3}$.
      select the minimum value of $\mu_n^{A=0}$, $\mu_n^{A=1}$, $\mu_n^{A=2}$ and $\mu_n^{A=3}$.
    end
    else vehicle user n cannot be connected to the UAV
      compute the value of payoff function $\mu_n^{A=2}$. **select** the minimum value of $\mu_n^{A=0}$, $\mu_n^{A=2}$.
    end
  end
  else vehicle user n cannot be connected to the MEC server
    if *vehicle user n can be connected to the UAV* **then**
      compute the value of payoff function $\mu_n^{A=1}$, and $\mu_n^{A=3}$.
      select the minimum value of $\mu_n^{A=0}$, $\mu_n^{A=1}$, and $\mu_n^{A=3}$.
    end
    else vehicle user n cannot be connected to the UAV
      choose the value of $\mu_n^{A=0}$.
    end
  end
  make $s_n \leftarrow$the value of A of the minimum value of payoff function.
  return s_n
until *there are no computation tasks need to be executed*;

efficiency. In light of the various possible situations in Table III, we next introduce the process of vehicle decision-making.

(a) Vehicle user n can establish a communication link with both the UAV and the MEC server

We introduce the situation that vehicle user n can establish a communication link with the UAV and the MEC server. Under this circumstance, the strategy set of user n includes executing the computation task locally, offloading the computation task to the MEC server directly (V-M), offloading the task to the UAV (V-U), and offloading the task to the MEC server through the relay of the UAV (V-U-M). Then, n will make the offloading decision based on the global information provided by the SDN controller. User n can obtain the optimal execution decision s_n^* by solving the following problem

$$\mu^{s_n}(n) = arg \min_{T_{I_n}, E_{I_n}} L_n^{s_n} = \lim_{\theta \to 0^+} \theta^{|s_n - 0|} L_{I_n,V}^{Loc} + \lim_{\theta \to 0^+} \theta^{|s_n - 1|} L_{I_n}^{V-U} + \lim_{\theta \to 0^+} \theta^{|s_n - 2|} L_{I_n}^{V-M} + \lim_{\theta \to 0^+} \theta^{|s_n - 3|} L_{I_n}^{V-U-M} \tag{30}$$

where $\lim_{\theta \to 0^+} \theta^{|s_n - d|}, d \in \{0, 1, 2, 3\}$ is the coefficient. When the execution decision $s_n = 0$ (i.e., local execution), $\lim_{\theta \to 0^+} \theta^{|s_n - 0|} = 1$ and other multiple items $\lim_{\theta \to 0^+} \theta^{|s_n - d|}, d \in \{1, 2, 3\}$ is 0 and $\mu^{s_n}(n) = L_{I_n,V}^{Loc}$.

Moreover, it is easy to prove that the Eq. (30) is a convex function.

$$\frac{\partial^2 L_n^{s_n}(T_{I_n}, E_{I_n})}{\partial^2 T_{I_n}} = 0, \frac{\partial^2 L_n^{s_n}(T_{I_n}, E_{I_n})}{\partial^2 E_{I_n}} = 0$$

According to the above proof, it can be concluded that Eq. (30) is a convex function. Consequently, it returns an optimal solution. At the same time, we can know that NE exists for the sequential game. The optimal strategy s_n^* for n is given by

$$s_n^* = \mu^{s_n}(n) = \begin{cases} arg \min_{T_{I_n}^{Loc}, E_{I_n}^{Loc}} L_n^{s_n} & if\ s_n = 0 \\ arg \min_{T_{I_n}^{V-U}, E_{I_n}^{V-U}} L_n^{s_n} & if\ s_n = 1 \\ arg \min_{T_{I_n}^{V-M}, E_{I_n}^{V-M}} L_n^{s_n} & if\ s_n = 2 \\ arg \min_{T_{I_n}^{V-U-M}, E_{I_n}^{V-U-M}} L_n^{s_n} & if s_n = 3 \end{cases} \tag{31}$$

(b) Vehicle user n only can establish a communication link with the UAV

We consider the condition that vehicle user n can only establish a communication link with the UAV. In this case, the strategy set of n will include executing the computation task locally, V-U, and V-U-M. Then, n will make the offloading decision based on the information provided by the SDN controller. User n can obtain the optimal execution decision s_n^* by solving the following problem

$$\mu^{s_n}(n) = arg \min_{T_{I_n}, E_{I_n}} L_n^{s_n} = \lim_{\theta \to 0^+} \theta^{|s_n - 0|} L_{I_n,V}^{Loc} + \lim_{\theta \to 0^+} \theta^{|s_n - 1|} L_{I_n}^{V-U} + \lim_{\theta \to 0^+} \theta^{|s_n - 3|} L_{I_n}^{V-U-M} \tag{32}$$

where $\lim_{\theta \to 0^+} \theta^{|s_n - d|}, d \in \{0, 1, 3\}$. In the meanwhile, we can

TABLE IV
SIMULATION PARAMETERS

Simulation Parameter	Value
Size of the simulation area	1000m×1000m
W	20MHz
α_V	4
N_0	-100dB
$F_{CPU}^{V,n}$	$\{0.5, 1.7, 0.8, 1.0\}$GHz
F_{CPU}^{UAV}	10GHz
F_{CPU}^{MEC}	50GHz
e_{CPU}^{V}	1u
e_{CPU}^{UAV}	1u
e_{SEND}^{V}	25u
e_{SEND}^{UAV}	50u

obtain the optimal strategy s_n^* for vehicle user n as

$$s_n^* = \mu^{s_n}(n) = \begin{cases} arg \min\limits_{T_{I_n}^{Loc}, E_{I_n}^{Loc}} L_n^{s_n} & if\ s_n = 0 \\ arg \min\limits_{T_{I_n}^{V-U}, E_{I_n}^{V-U}} L_n^{s_n} & if\ s_n = 1 \\ arg \min\limits_{T_{I_n}^{V-U-M}, E_{I_n}^{V-U-M}} L_n^{s_n} & if s_n = 3 \end{cases} \tag{33}$$

(c) Vehicle user n can establish a communication link with the MEC server only, but not with UAV

We consider the condition that vehicle user n can only establish a communication link with the MEC server here. In this case, the strategy set of n includes performing the computation tasks locally, and V-M. n will make the offloading decision based on the information given by the controller. Then, n can obtain the optimal execution decision s_n^* by solving the following problem

$$\mu^{s_n}(n) = arg \min_{T_{I_n}, E_{I_n}} L_n^{s_n} = \lim_{\theta \to 0^+} \theta^{|s_n - 0|} L_{I_n, V}^{Loc} + \lim_{\theta \to 0^+} \theta^{|s_n - 2|} L_{I_n}^{V-M} \tag{34}$$

where $\lim\limits_{\theta \to 0^+} \theta^{|s_n - d|}, d \in \{0, 2\}$. Then, we can obtain the optimal strategy s_n^* for vehicle user n as

$$s_n^* = \mu^{s_n}(n) = \begin{cases} arg \min\limits_{T_{I_n}^{Loc}, E_{I_n}^{Loc}} L_n^{s_n} & if\ s_n = 0 \\ arg \min\limits_{T_{I_n}^{V-M}, E_{I_n}^{V-M}} L_n^{s_n} & if\ s_n = 2 \end{cases} \tag{35}$$

(d) Vehicle user n cannot establish a communication link with the MEC server or the UAV

We consider the condition that vehicle user n who is out of the communication range of the MEC server and the UAV here. In this circumstance, n cannot offload the computation task I_n to the UAV or the MEC server to execute the computation task. Therefore, the strategy of n can only execute the task locally. We can obtain the optimal strategy s_n^* for n as

$$s_n^* = arg \min_{T_{I_n}^{Loc}, E_{I_n}^{Loc}} L_n^{s_n},\ s_n = 0 \tag{36}$$

Based on the above statement, we design the UVCO algorithm that can be seen in Algorithm 1. Multiple legacy vehicle users can generate a set of computation tasks with time sensitivity. We set the time-sensitivity equals to the weight of the cost of execution time in the payoff function. The objective of the UVCO algorithm is to achieve mutually satisfactory decision-making and optimizing the cost (i.e., execution time and energy consumption) simultaneously. In order to synchronize the time of each device, the time signal synchronization of MEC server is adopted. When a task generated by the vehicle user needs to be executed, the vehicle user can make a satisfactory decision mutually according to Algorithm 1. The algorithm terminates when every vehicle user completes the decision and reaches equilibrium.

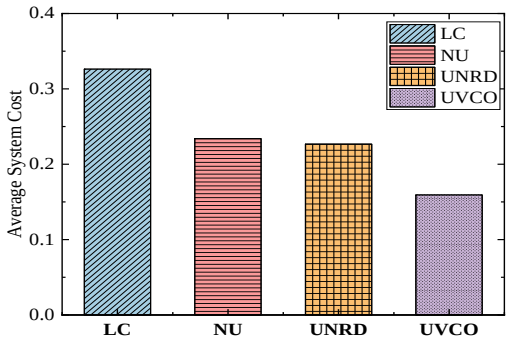

Fig. 2. Average system cost of each strategy

IV. EXPERIMENTS

In this section, we will present our experimental results. To evaluate our proposed UVCO algorithm, we develop a simulation platform to meet the requirements of vehicular computation offloading. We first capture a real map by 1000 m × 1000 m, where the buildings are modeled according to the map topology. SUMO is applied to simulate the trajectory of $N = 30$ vehicles on the map, in which the MEC server is located on the left-most side of the map whose communication range is large enough to cover the map region. The trajectory of the UAV is a circle with coordinates (500, 500) as its center and 100 m as its radius. The altitude of the UAV is fixed at 300 m, in which it runs at a uniform speed of 20 m/s. For the wireless communications, the channel bandwidth is W=20 MHz; the background noise is set as N_0=-100 dB; and the transmission power is $R_W iFi$=100 mWatts. During the movement of the vehicle, there will be computation tasks I need to be executed after one computation task of a vehicle is solved, the next computation task will be generated at an interval of 1 second. The priority of the computation task equals to γ_n^T in the utility function and γ_n^E is based on the execution time weight which is $\gamma_n^E = 1 - \gamma_n^T$. We consider F_{CPU}^{UAV} as 10 GHz, and F_{CPU}^{MEC} as five times more powerful than the UAV. We assume that the computing power of the vehicles are poor, by selecting a value from a set as $F_{CPU}^{V,n} \in \{0.5, 0.7, 0.8, 1.0\}$ randomly. The transmission rate of the vehicle to the UAV or the MEC server depends on how many vehicles transmit data simultaneously which can be calculated from Eq. (1) and Eq. (2). Since there is only

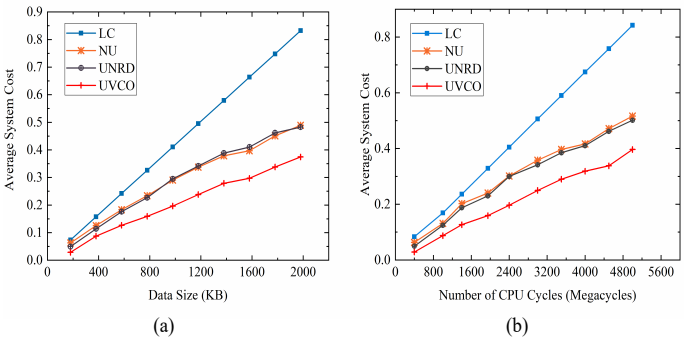

Fig. 3. The average system cost with the change of the offloaded data size and the number of CPU cycles

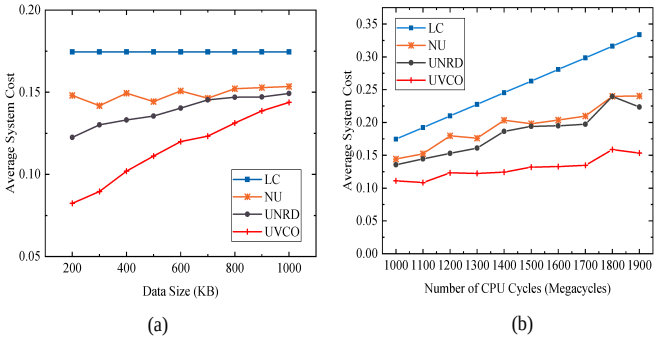

Fig. 4. The impact of data size and CPU cycles on average system cost

one UAV in the scenario, the transmission power of the UAV is fixed at 20 Mbps. Table IV shows the parameter setting of our vehicular computation offloading and simulator.

We evaluate the performance of the proposed UVCO algorithm in cost optimization. The evaluation method is compared with three other strategies. The first one is all vehicles can only execute computation tasks locally (LC). Second, all vehicles can offload computation tasks to the MEC server through the MEC server or execute computation tasks locally, while there is no UAV (NU). In the third strategy, there is a UAV, which has a certain computing power. Vehicles can randomly choose to offload the computation tasks to the UAV or the MEC server to execute the task instead of the vehicles. However, the UAV cannot be used as a relay node (UNRD). If the vehicle cannot establish a communication connection with the UAV and the MEC server, the vehicle user can only choose to execute the task locally.

As shown in Fig. 2, we mainly study the average cost of the system in these different strategies to deal with computation tasks. The results show that the system average cost of all local computing is the highest while the UVCO is the lowest. In these strategies, vehicles generate the same size of computation tasks, where the size is 780 KB and the number of CPU cycles is 1857 Megacycles. When all other parameters are the same, UVCO outperforms the other three strategies LC, NU and UNRD with achieving 51.19%, 31.94% and 29.72% gain in terms of the parameters to the vehicle users. In this case, the utility of the system cost is minimized to make optimal decisions for achieving NE. Without acting as a relay node, UNRD is less costly than the strategies LC and NU. This proves that the involvement of the UAV is necessary indeed which can reduce the average cost of the system.

In Fig. 3, we investigate the impact of offloaded data size (O_i) and the number of CPU cycles (C_i) changed simultaneously on the system average cost in the four strategies. In the simulation, we change the size of the offloaded data, and the number of CPU processing cycles. Fig. 3 shows that as the size of the offloaded data and the computational complexity in terms of the number of CPU cycles increase, the average system cost (ASC) of all strategies climbs up to varying degrees. Results show that our UVCO algorithm outperforms other three strategies LC, NU and UNRD achieving from 42.4% to 55.01% gain in terms of the ASC. When the size of offloaded data increases, the transmission time and system energy consumption enlarge. Also, with the increase in the number of CPU processing cycles, the complexity of the computation increased and the cost also rises. In addition, with the same parameters, the ASCs of NU and UNRD are closed, since the relative energy consumption of UAV is larger, and the advantages of execution time cost and the disadvantages of energy consumption are offset. The LC strategy climbs up linearly as the highest ASC. The reason is that the local computation depends on the computation frequency of the vehicle only.

Fig. 4(a) illustrates the effect of data size on ASC in computation offloading. We fix the computational complexity represented by the number of CPU cycles (C_i), then change the size of the offloaded data (O_i). With the increment of the size of offloaded data, all of NU, UNRD, and UVCO show an upward trend. UVCO outperforms the other three strategies LC, NU and UNRD achieving from 1.46% to 52.82% gain in terms of the ASC. Nevertheless, LC has not received any impact, in which the reason is that all vehicles computation tasks are locally executed. Therefore, the change of offloaded data size has no effect on LC. As shown in Fig. 4(a), the ASC of UNRD is much lower than that of NU as the offloaded data size is small. However, with the increase in the data size, the ASC of the two strategies tend to be close. The main reason is that the UAV joins the system, which increases the throughput of the system, but it also lifts up the cost consumption. For offloading computation tasks, the larger the size of the data can be offloaded, the more cost it will consume.

To evaluate the impact of the computation size on computation offloading, we implement the simulations with a different number of C_i required for performing computation tasks, and fixed O_i. In Fig. 4(b), we observe that the overall system cost of four strategies keeps climbing as the number of CPU cycles increase. Since the increment of the number of CPU cycles for performing computation tasks, the cost of computing tasks for devices rises accordingly. UVCO outperforms the other three strategies LC, NU, and UNRD by achieving from 18.01% to 52.44% gain in terms of the ASC. The system cost of NU is slightly higher than that of UNRD. Therefore, it can be concluded that the existence of the UAV can effectively reduce the system cost. To this end, UVCO can balance the cost of the execution time and the energy consumption to achieving

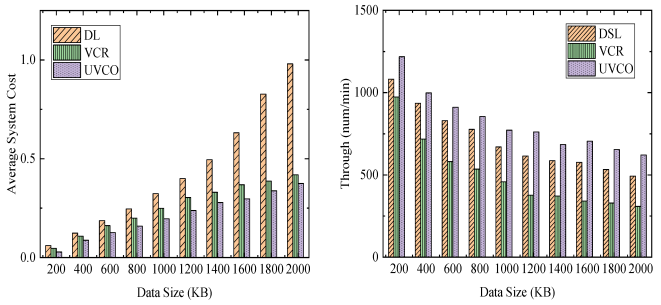

Fig. 5. The performance comparison in terms of average system cost and throughput

NE.

In the following, we compare the proposed UVCO with the VCR algorithm [28] and deep learning (DSL) algorithm [19] in terms of the ASC and throughput. Combining the analysis from Fig. 5(a) and (b), we can draw that UVCO can calculate the optimal offloading decision for different data sizes in terms of ASC and throughput (i.e., the number of computational tasks completed per minute) under the same scenario. We set the scenario with 30 moving vehicles and generating the computation tasks. Three algorithms are adopted to make the offloading decision in order to observe the impact of increasing data size on ASC and average system throughput per minute. Fig. 5(a) illustrates that as the increase of data size, UVCO achieves the lower cost from 39.07% to 61.78%. Furthermore, in Fig. 5(b), UVCO can provide more throughput than its counterparts varying from 11.17% to 62.44%. Benefited from the design of global information provided by the controller, it can save a lot of time to collect information. Hence, the proposed UVCO algorithm makes the optimal decision in order to achieve NE and minimize system cost in our scenario. Since the DSL algorithm makes the offloading decisions for a single user only, it takes no account of the interaction and cooperation between multiple users. Also, in existing work [28], once an intelligent device executes an offloaded computation task, it will broadcast the occupied signal and cannot be offloaded until the computation task is completed. In this context, the intelligent device will be idle within the time from the completion of the computation task to the transmission of the next computation task. In our proposal, the waiting queue of offloading tasks perfectly refrains from the resource waste caused by the above issue. Regarding the effectiveness, UVCO is an optimal algorithm for offloading decision-making in the proposed scenario.

V. Conclusion

In this paper, we consider the dynamic movement of multiple legacy vehicles in the scenario with dense buildings and scarce infrastructure. In our solution, we employ a UAV with certain intelligent computing power to assist the legacy vehicles in executing or relaying time-sensitive and resource-sensitive computation tasks and design the UVCO algorithm with the idea of game theory. According to the offloading parameters provided by the SDN controller, UVCO minimizes the system cost by formulating mutually satisfactory offloading decisions. Moreover, in order to fully utilize the resources in the system, a waiting queue is set in the UAV and the MEC server for the offloaded computation tasks to improve efficiency. Experimental results show that the proposed algorithm has outstanding performance in terms of ASC and system throughput. UVCO reduces the ASC by 55.01%. Furthermore, compared with its counterparts, UVCO achieves as high as 61.78% reduction in terms of the ASC. For the system throughput, UVCO achieves up to 62.44% improvement compared with the related works.

Acknowledgment

This paper is partly supported by the National Natural Science Foundation of China (61701322), the Young and Middle-aged Science and Technology Innovation Talent Support Plan of Shenyang (RC190026), the Liaoning Natural Science Foundation (2020-MS-237), and the Liaoning Provincial Department of Education Science Foundation (JYT19052).

References

[1] K. Abboud, H. A. Omar, and W. Zhuang, "Interworking of dsrc and cellular network technologies for v2x communications: A survey," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9457–9470, Dec 2016.

[2] F. Abbas, P. Fan, and Z. Khan, "A novel low-latency v2v resource allocation scheme based on cellular v2x communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 6, pp. 2185–2197, June 2019.

[3] N. Lin, L. Fu, L. Zhao, G. Min, A. Al-Dubai, and H. Gacanin, "A novel multimodal collaborative drone-assisted vanet networking model," *IEEE Transactions on Wireless Communications*, vol. PP, pp. 1–1, 04 2020.

[4] H. Li, M. Dong, and K. Ota, "Control plane optimization in software-defined vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 10, pp. 7895–7904, Oct 2016.

[5] G. Luo, S. Jia, Z. Liu, k. Zhu, and L. Zhang, "sdnmac: A software defined networking based mac protocol in vanets," pp. 1–2, June 2016.

[6] Q. Zheng, K. Zheng, H. Zhang, and V. C. M. Leung, "Delay-optimal virtualized radio resource scheduling in software-defined vehicular networks via stochastic learning," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 10, pp. 7857–7867, Oct 2016.

[7] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani, "Software-defined networking for rsu clouds in support of the internet of vehicles," *IEEE Internet of Things Journal*, vol. 2, no. 2, pp. 133–144, April 2015.

[8] L. Zhao, G. Han, Z. Li, and L. Shu, "Intelligent digital twin-based software-defined vehicular networks," *IEEE Network*, pp. 1–7, 2020.

[9] C. Lin, G. Han, X. Qi, M. Guizani, and L. Shu, "A distributed mobile fog computing scheme for mobile delay-sensitive applications in sdn-enabled vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5481–5493, 2020.

[10] J. Cheng, J. Cheng, M. Zhou, F. Liu, S. Gao, and C. Liu, "Routing in internet of vehicles: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2339–2352, Oct 2015.

[11] G. Han, H. Wang, X. Miao, L. Liu, J. Jiang, and Y. Peng, "A dynamic multipath scheme for protecting source-location privacy using multiple sinks in wsns intended for iiot," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5527–5538, 2020.

[12] G. Min and M. Ould-Khaoua, "A performance model for wormhole-switched interconnection networks under self-similar traffic," *IEEE Transactions on Computers*, vol. 53, no. 5, pp. 601–613, 2004.

[13] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE Communications Surveys Tutorials*, vol. 13, no. 4, pp. 584–616, Fourth 2011.

[14] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, April 2015.

[15] C. Wu, Z. Liu, F. Liu, T. Yoshinaga, Y. Ji, and J. Li, "Collaborative learning of communication routes in edge-enabled multi-access vehicular environment," *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2020.
[16] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, October 2016.
[17] M. Deng, H. Tian, and X. Lyu, "Adaptive sequential offloading game for multi-cell mobile edge computing," pp. 1–5, May 2016.
[18] J. Zhao, L. Wang, K. Wong, M. Tao, and T. Mahmoodi, "Energy and latency control for edge computing in dense V2X networks," *CoRR*, vol. abs/1807.02311, 2018.
[19] S. Yu, X. Wang, and R. Langar, "Computation offloading for mobile edge computing: A deep learning approach," pp. 1–6, Oct 2017.
[20] X. Hu, K. Wong, K. Yang, and Z. Zheng, "Uav-assisted relaying and edge computing: Scheduling and trajectory optimization," *IEEE Transactions on Wireless Communications*, vol. 18, no. 10, pp. 4738–4752, Oct 2019.
[21] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 587–597, 2018.
[22] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Drone small cells in the clouds: Design, deployment and performance analysis," pp. 1–6, Dec 2015.
[23] M. Messous, A. Arfaoui, A. Alioua, and S. Senouci, "A sequential game approach for computation-offloading in an uav network," pp. 1–7, Dec 2017.
[24] H. Ghazzai, H. Menouar, and A. Kadri, "On the placement of uav docking stations for future intelligent transportation systems," pp. 1–6, June 2017.
[25] P. Pourbaba, K. B. S. Manosha, S. Ali, and N. Rajatheva, "Full-duplex uav relay positioning for vehicular communications with underlay v2v links," pp. 1–6, April 2019.
[26] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, June 2016.
[27] E. Kalantari, M. Z. Shakir, H. Yanikomeroglu, and A. Yongacoglu, "Backhaul-aware robust 3d drone placement in 5g+ wireless networks," pp. 109–114, May 2017.
[28] Z. Wang, Z. Zhong, D. Zhao, and M. Ni, "Vehicle-based cloudlet relaying for mobile computation offloading," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 11 181–11 191, Nov 2018.
[29] A. Alioua, S. Senouci, S. Moussaoui, H. Sedjelmaci, and M. Messous, "Efficient data processing in software-defined uav-assisted vehicular networks: A sequential game approach," *Wireless Personal Communications*, vol. 101, pp. 2255–2286, 2018.

Liang Zhao [M] is an associate professor at Shenyang Aerospace University, China. He received his PhD degree from the School of Computing at Edinburgh Napier University in 2011. Before joining Shenyang Aerospace University, he worked as an associate senior researcher in Hitachi (China) Research and Development Corporation from 2012 to 2014. He has published more than 70 papers. His research interests include VANETs, SDVN, FANETs and WMNs.

Kaiqi Yang received the B.S. degree in Computer Science and Technology from Shenyang Aerospace University, China. Currently she is a Master student in the School of Computer Science at Shenyang Aerospace University. Her research interests mainly include VANETs, FANETs, SDVN and Computation offloading.

Zhiyuan Tan received the Ph.D.degree in computer systems from the University of Technology Sydney, Ultimo, NSW, Australia, in 2014. He was a Postdoctoral Researcher of cybersecurity with the University of Twente, The Netherlands, from 2014 to 2016, and a Research Associate with the University of Technology Sydney, in 2014. He is currently a Lecturer of cybersecurity with the School of Computing, Edinburgh Napier University, U.K. His research interests include cybersecurity, machine learning, pattern recognition, data analytics, virtualization, and cyber-physical systems.

Xianwei Li received his MS degree from Hunan University, Changsha, China in 2010 and PhD degree from Waseda University, Tokyo, Japan in 2019. Since July 2011, he has been with the School of Information Engineering, Suzhou University. His research interests include mobile edge cloud computing, Internet of Things, machine learning and big data.

Suraj Sharma was a Project Associate with NIT Rourkela, India. He has been an Assistant Professor with the Department of Computer Science, International Institute of Information Technology Bhubaneswar, India, since 2012, where he is the Director of the IoT and Cloud Computing Laboratory and working with many Ph.D. and M.Tech. scholars. He has published over 35 research papers in the reputed international journals and conferences. His current research areas include Internet of Things (IoT), wireless sensor networks, security, IoV, and cloud computing.

Zhi Liu [SM] received the B.E., in computer science and technology from the University of Science and Technology of China, China, in 2009 and Ph.D. degree in informatics in National Institute of Informatics and The Graduate University for Advanced Studies (Sokendai) Tokyo, Japan. He is currently an Assistant Professor at Shizuoka University and an adjunct researcher at Waseda University, Japan. His research interest includes wireless networks, video/image processing and transmission, and he is a senior member of IEEE.